PHPの教え方と学び方のコツ

by 古庄道明(がる/gallu)

自己紹介

- 古庄道明と申します
- 「がる(gallu)」というハンドルでふらついております
- 本職は技術者です。現役プログラマーやってます
  - バックエンド系なので、インフラとかDBとか運用とかも一通り
    - 直近は、ゲームのサーバサイドをやってます
  - 最近はPM業とか技術支援(サポート)とかも多いですねぇ
  - 「ヒアリングして技術提案」なんてこともさせていただいてます
- 教育も色々と携わっております(専門学校講師とか)
- 「PHP8技術者認定上級試験」「PHP8技術者認定初級試験」の作成を担当させていただいてます
  - 問題集も(ベータですが)出ました!!
- コラムとかblogとかeラーニングの教材とかで技術系の文章を書くこともちらほら

今日のお題

- 「PHPの教え方と学び方のコツ」という事で
  - 新人/初心者が学習するための道のりの一例をたどっていければ、と思います
  - 「新人/初心者を教える」人もいるかと思うので、「教える事の初心者」さんもいる前提で、教え方やポイントも見ていきます
  - 一方で「独学」の人もいるかと思うので、「独学の時のポイント」も押さえていきましょう
    - とはいえやっぱり「完全な独学」は少しシンドイのですが……
- 去年にお話をした内容とかぶる箇所もありますが、追加もいろいろあります!!

学習環境の準備

- 学習環境ですが、主に以下が必要になります
- PC(OS)
  - パソコンがないと厳しい事が多いです。OSはお好みで
- エディタ / IDE
  - 「プログラムを書く」直接のツールです
    テキストエディタでもよいですが、もうちょっと「プログラムを書く事にマッチした」エディタだと、はかどります
- 実行環境
  - 「自分のPC内に構築」するか「お外にサーバを用意するか」
  - この環境によって「書いたプログラムをどうやって実行環境に渡すか」の手段も変わってきます

- 実行環境をかみ砕くと
  - httpd
    - あまり意識しない所ですが「ないと動かない」です
  - PHP本体
    - 主に「どのバージョンで学習するか」
  - RDB
    - 序盤はなくてもよいですが、比較的早いタイミングで必要になります

&

- 「ある程度まとまった時間」

プチ余談「まとまった時間」？

- 「8時間集中しろ」とは言わないのですが
  - そもそも人間、8時間集中とか無理ですし
- 「隙間時間の5分を積み重ねる」のは、序盤の頃はちょっと効率が悪いように思います
- できれば最低でも15～30分くらい、可能なら1時間くらいは「集中できる」とよいです
- あと「月に数回」も「前回の内容を忘れやすい」ので、できたら「週に数回」くらいは学習できる時間が確保できるとよいと思います
  - 基本ができてくると「月に数回」でも学習効果が出ます！

学習環境を準備する大切さとハードル

- 学習環境の準備は大切になります
  - プログラムは「書いて」「動かして」「エラー出して」「修正して」を繰り返さないと、なかなか身につきません
  - そのため「動かすことができる」学習環境の構築は、必須事項になります
- 一方で「学習環境の準備でつまづく」ケースも割と耳に目にするので、ここのハードルを「いかに下げていくか」は大切です

新人 / 初心者(以下新人)

- 学習環境の構築は学習のための「必須条件」です
- が、実は「結構に難易度もハードルも高くなりやすい(常に高いわけじゃない)」ので、ポイントとかコツとか必要
  - ですがそれがわかる人は新人とか初心者とかじゃない
- というわけで、先輩に「頼って」しまいましょう
  - 多分きっとなんかノウハウや手順書があるはずです!!
- あと、できるだけ早く「質問できる環境」を整えると吉！
  - 「質問の仕方」はちゃんと学習しておきましょう！
    - 書籍) アプレンティスシップ・パターン

先輩向け

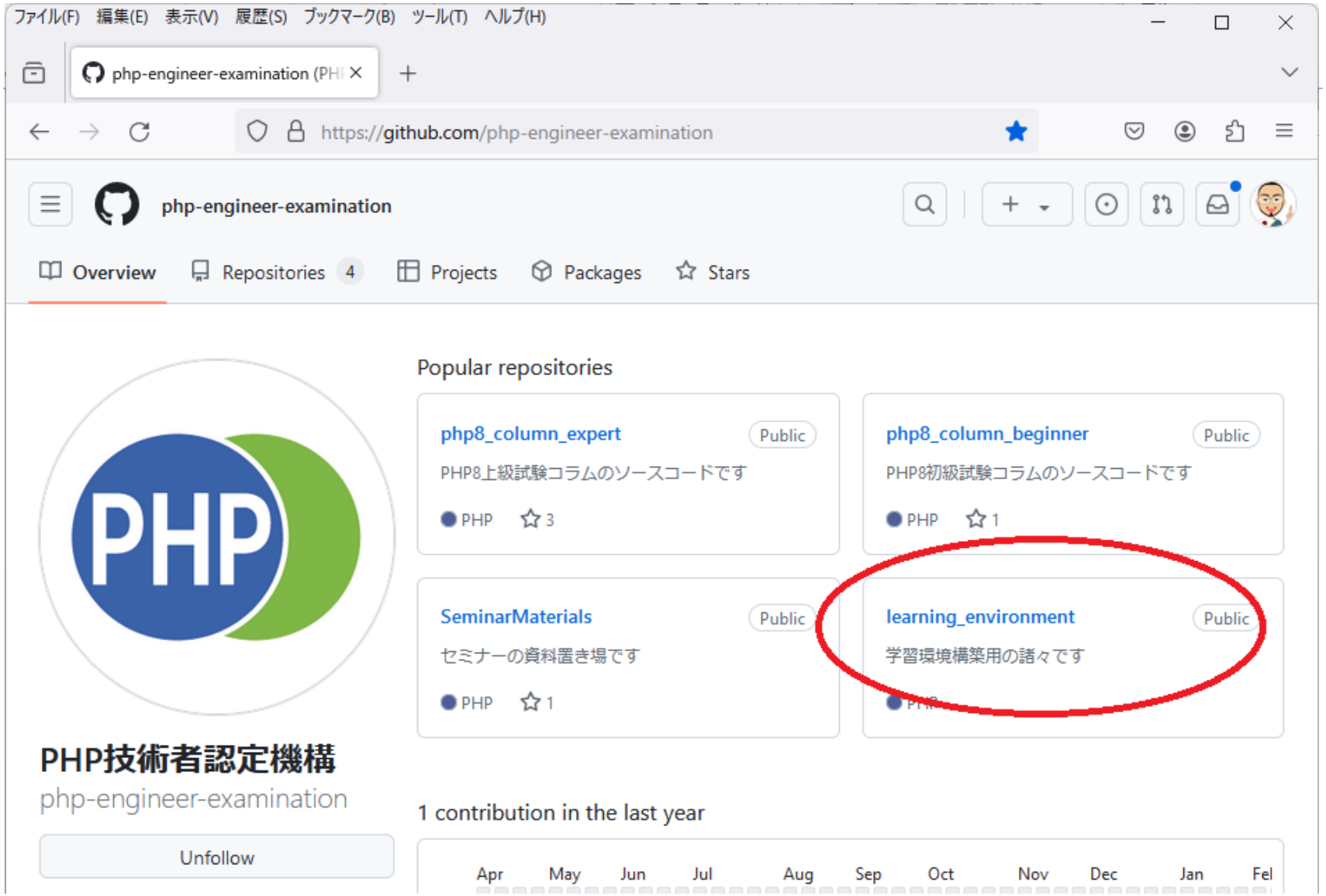
- 新人さんに学習環境を整える(手順を作る)時のポイントは「わかりやすく」「間違えにくく」です
- 会社さんで用意する場合、ほとんどの場合「社内で使っている環境」があるかと思うので、それに沿った形にするのが一番無難です
- 導入の手順だけは「丁寧にきめ細やかに」
  - 「新人さんが躓いた」時は「マニュアル改善のチャンス」だと思うとよいです
  - 新人さんが「質問しやすい」ようにするためにもいろいろ工夫をしましょう
    - 細かく「チェックポイント」
    - 折々の「雑談」

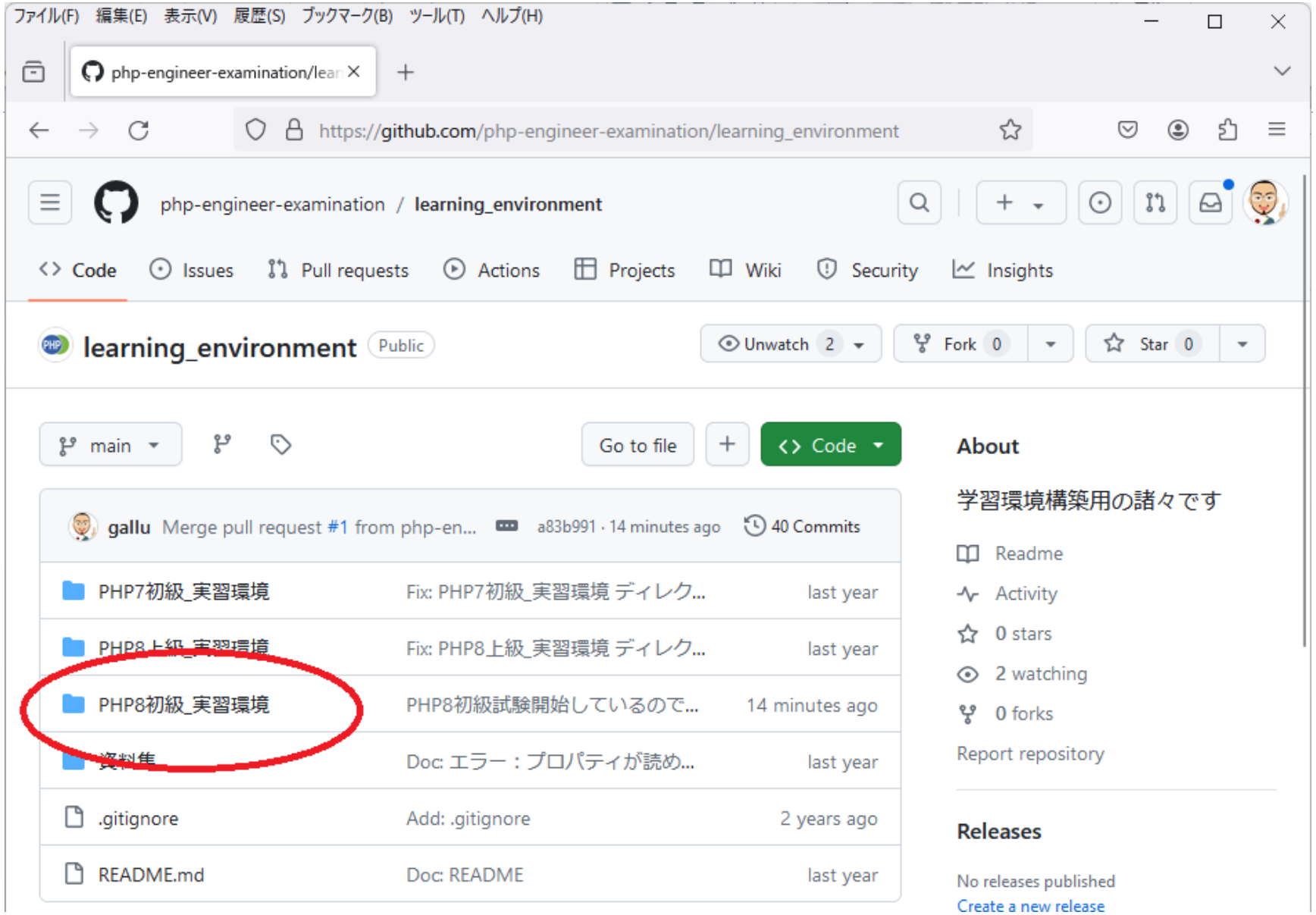
独学

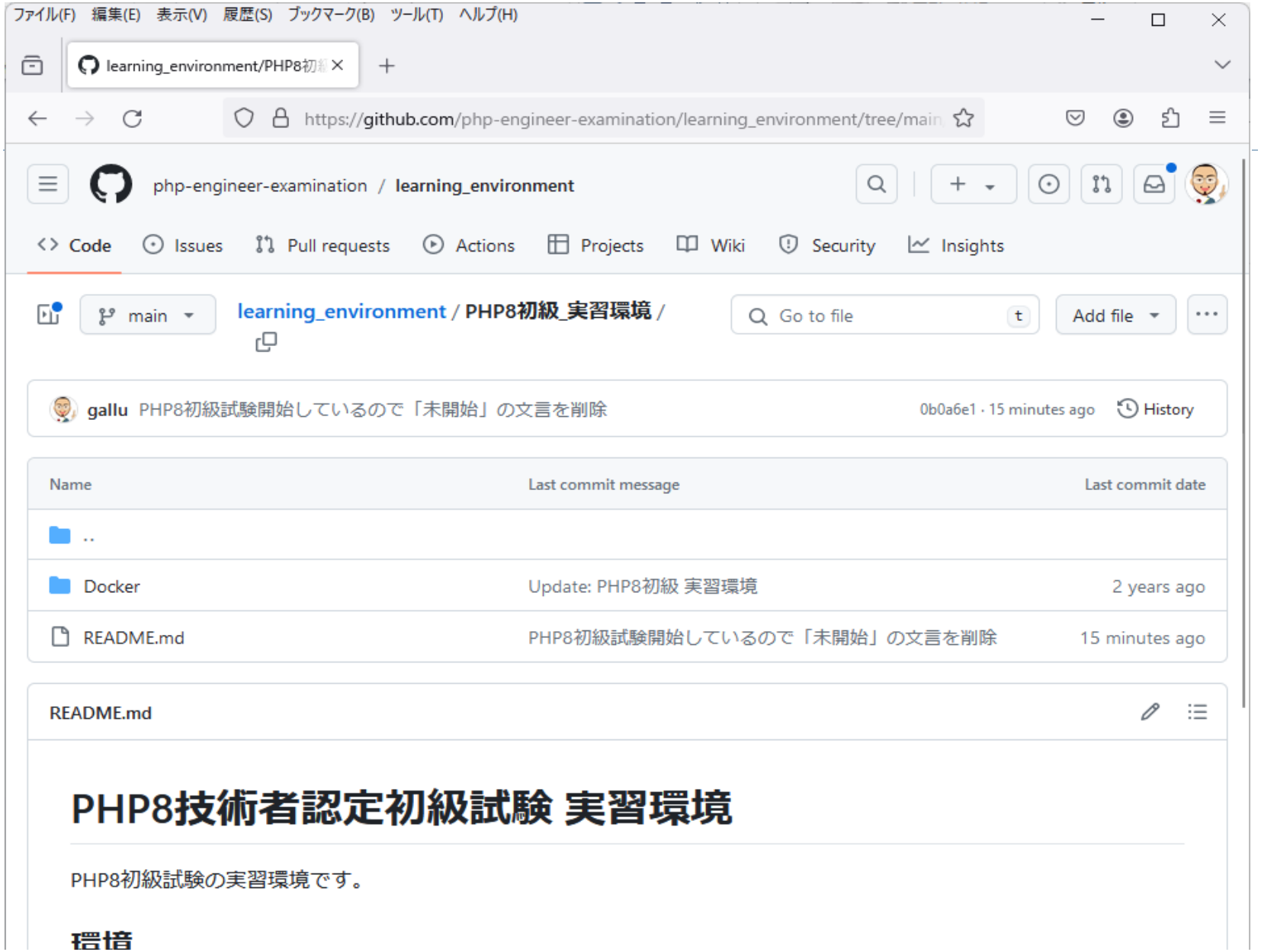
- 独学は「一人で好きなときに好きなペースでできる」と「独りで全部片付けないといけない」が共存します
- 書籍やEラーニングベースだと思うので基本は「書いてある内容に可能な限り忠実に」をベースにするとよいです
  - なので、書籍にしてもEラーニングにしても「あんまり古いもの」は環境構築のハードルも上がりやすいので、できるだけ「ある程度(以上)新しいもの」で学習するとよいでしょう
- もし万が一「ネットなどで"質問できる人(≒メンター)"」を捕まえたら、速やかにお勧めを質問しましょう!

- というのを踏まえた上で、独学用に軽くポイントを

- PC(OS)
  - 本当にお好みで。Mac人気ですがおいちゃんはWinです
- エディタ / IDE
  - VSCodeが「無料」「人気」なので、お勧めです
    - 人気だと「ググって調べた時に情報がたくさんある」ので便利!!
- 実行環境(+PHP+httpd+RDB)
  - 独学だと PHP Open Textbook(https://github.com/php-engineer-examination/)にある learning_environment 内「PHP8初級_実習環境」をお勧めします!!
    - ダイレクトマーケティング
  - 使い方のより丁寧な説明、近々に対応します！
    - ……って言って1年が経過していますねすみません orz


ファイル(F) 編集(E) 表示(V) 履歴(S) ブックマーク(B) ツール(T) ヘルプ(H)
php-engineer-examination (PH
https://github.com/php-engineer-examination
php-engineer-examination
Overview
Repositories 4
Projects
Packages
Stars
Popular repositories
php8_column_expert
Public
PHP8上級試験コラムのソースコードです
PHP 3
php8_column_beginner
Public
PHP8初級試験コラムのソースコードです
PHP 1
SeminarMaterials
Public
セミナーの資料置き場です
PHP 1
learning_environment
Public
学習環境構築用の諸々です
PHP
PHP技術者認定機構
php-engineer-examination
Unfollow
1 contribution in the last year
Apr May Jun Jul Aug Sep Oct Nov Dec Jan Fel


ファイル(F) 編集(E) 表示(V) 履歴(S) ブックマーク(B) ツール(T) ヘルプ(H)
php-engineer-examination/lear
https://github.com/php-engineer-examination/learning_environment
php-engineer-examination / learning_environment
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
learning_environment
Public
Unwatch 2
Fork 0
Star 0
main
Go to file
Code
gallu Merge pull request #1 from php-en... a83b991 · 14 minutes ago 40 Commits
PHP7初級_実習環境 Fix: PHP7初級_実習環境 ディレク... last year
PHP8上級_実習環境 Fix: PHP8上級_実習環境 ディレク... last year
PHP8初級_実習環境 PHP8初級試験開始しているので... 14 minutes ago
資料集 Doc: エラー：プロパティが読め... last year
.gitignore Add: .gitignore 2 years ago
README.md Doc: README last year
About
学習環境構築用の諸々です
Readme
Activity
0 stars
2 watching
0 forks
Report repository
Releases
No releases published
Create a new release


ファイル(F) 編集(E) 表示(V) 履歴(S) ブックマーク(B) ツール(T) ヘルプ(H)
learning_environment/PHP8初
https://github.com/php-engineer-examination/learning_environment/tree/main
php-engineer-examination / learning_environment
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
main
learning_environment / PHP8初級_実習環境 /
Go to file
t
Add file
gallu PHP8初級試験開始しているので「未開始」の文言を削除
0b0a6e1 · 15 minutes ago
History
Name
Last commit message
Last commit date
..
Docker
Update: PHP8初級 実習環境
2 years ago
README.md
PHP8初級試験開始しているので「未開始」の文言を削除
15 minutes ago
README.md
PHP8技術者認定初級試験 実習環境
PHP8初級試験の実習環境です。
環境

日本語で簡単に補足

- https://github.com/php-engineer-examination/learning_environment を git clone
  - ダウンロード先を「基準ディレクトリ」と仮称
- 基準ディレクトリ/PHP8初級_実習環境/Docker に移動
- (Dockerが立ち上がっていることを確認)
  - Windowsだと「Docker Desktop」が楽です
- docker-compose up -d
- http://localhost/test.php で接続確認
- プログラムは以下のディレクトリに作成(test.phpがある)
  - 基準ディレクトリ/PHP8初級_実習環境/Docker/html
- 終わる時は docker-compose down

学びはじめ

- 何事も、一番はじめが「とても大切」ですが「つまづきやすい」所でもあります
- 接客業には「15秒ルール」なんてのがございますが、プログラミングの学習も「初手でひっかかりまくる」といろいろと心証が悪化します
- 無理なく丁寧にコツコツとやっていく事をお勧めします
  - 1) 石橋を叩いて渡るように丁寧に
  - 2) 加速してコケてコケ方を覚えながら
  - 3) 先走ってわかんなくなったら巻き戻しながら
    - この辺はお好みで

- まずは「写経」なんて言われますが、実際「書籍やサンプルのコードを書き写して動かしてみる」のが1番はじめになろうかと思います
- 「自分が書いたコードが動く」というのは、簡単なものであれ、やはり楽しいものかと

- ……思うのですが、その前に

「記号の打ち方を確認しよう」

- 通常の文字についてはある程度「なれて」おきましょう
  - 「タイピングソフト」とかも便利ですので、適宜よしなに
- 問題は記号。プログラミングでは、〃とか；とか＜とか？とかその他いろいろ、普段普通に文字を打っていると「滅多に使わないよねぇ」な記号がやたらに出てきます
  - まず＜＞＄？；”‘／＝！＃＊－が入力できるようになりましょう
- この辺の入力の仕方を、事前に確認しておきましょう
  - 手前味噌恐縮ですが、よろしかったら https://www.phpexam.jp/archives/3124 古庄親方の初級試験コラム#001「プログラミング学習の準備運動」を

それでは、コードを書いていきましょう!

- 俗に「写経」なんて言い方をしますが
- まずは、先輩や本が出してくる「サンプルコード」を、そのまま打ち込んで、実行してみましょう!!
  - そのために「書いたコードの動かし方」をまずは理解把握しましょう!!

新人

- プログラミングは、ある程度までは「質量転化」
- また、プログラミングは「書いて」「動かす」がセット
- なのでまずは「大量に書いて大量に動かす」
- ポイントは「ちょっと書いたらとっとと動かす」の繰り返し
- どんどん書いて、まずは「プログラムを書く」事へのハードルを、どんどん下げていきましょう！

先輩

- 特に始めの頃は「細かく」が有効になりやすいです
- なので「すぐに書けそうなサンプルや例題や問題」を山盛りで用意するのがよいです
  - 準備時間とかたくさん必要になるので、そのための時間捻出をしっかりと意識しておきましょう(業務との兼ね合いとかあるでしょうし)
  - 「変数の内容をちょっと変える」くらいのバリエーションでOK!
- 書いている時も、可能な限り「こまめに見てあげる」とモチベーションが上がりやすいようです
  - 「褒める」一言を忘れずに！

独学

- 書籍等のサンプルを書いていく……事になるのですが、量的にどうしても「そこまで大量ではない」んですよね
- なので、はじめの頃は「前回書いたコードを消して(本当に消すともったいないので、どこか別の所に移動させて)改めて「同じコード」を書き直してみる」とかで少しだけ量を稼げます
- はじめの頃、2～3回くらいでいいのでやっておくと、「前回よりスムーズにかけた」など、成長が見て取れるんじゃないか、と思います

エラーが出た!!

- 「書籍やお手本の通りに書く」事ができれば、エラーが出る事はありません
  - ……多分
    - 書籍等が古いと「新しいバージョンはエラー等が出る」事があります
    - 「書籍のコードにミスがある」ケースも。正誤表を確認してみましょう
- でもまぁ「エラー」は出ます。にんげんだもの
  - 具体的には「打ち間違い」をします
- エラーをそのままにすると学習にならないので……さてどうしましょうか？
  - エラーの修正は、それ自体がよい学習でもあります

新人

- はじめの頃は「動かなかった」以上の事はわからないと思います
  - あるいは「こうなるよ」とは違う結果になった、くらい
- 先輩やメンターがいるんなら、躊躇なく速やかに質問しましょう
- 1度2度くらいは「見直してみる」「書き直してみる」も有効ですが、あんまりここで引っかかりすぎると「心が削れる」ので、早め早めに質問してしまいましょう
  - 何回か経験すると「あれ? 前もあったな? ここかな?」って見当がつき始めます

先輩

- ある意味、腕の見せ所です。比較的単純なtypoが多いか、とは思うのですが、それを見つけるには、割とスキルとか慣れとかが必要だったりします
  - 文末の；忘れ、〃等の対応のミス、<?とphpの間にスペース、コードの中に全角スペース、コードの文字が全角、等
- 基本は「エラーメッセージ見て行を特定する」普段のデバッグをやればよいのですが、「普段遭遇しないミス」とかも割とあるので、先入観を捨てていくとよいでしょう

- 先輩の腕の見せ所ですガンバレ！

独学

- ……一番つまづきやすいところで、正直ここを「独学だけでどうにかする」のは、割とむずかしいです
  - なのでネット等で「メンター」を捕まえる事をお勧めします
- つまった時には「いっかい、休む」「その後、新しいファイルで改めて書いてみる」と、解決する、時もあります
- もしネットで質問をする時は「プログラムを全部コピペで貼り付ける」「エラーメッセージもコピペで貼り付ける」と、ワンチャンある、かもしれません
  - ChatGPTなどのAIを使ってもよい、のですが、「必ず正解が出てくる」とも限らないのが難しいところです……
  - が、独学ならワンチャンあるので、試してみましょう!!
    - 「修正されたコードのコピペ」は避けたほうがよいです
    - ちゃんと「どこが間違っているか」を理解して、自分で修正しましょう!!

- ここまでで、まず「一番はじめ」の頃の学習の「大切な基本」について述べていきました
- まぁぶっちゃけ序盤は「まず、プログラミングになれる」のが最優先かと思います
  - プログラム書いた！
  - プログラム動いた！

エラーが出た その2

- エラーとは、長い付き合いになります(笑
- なので、はじめは「怖い」エラーとも、だんだん、仲良くなる必要が出てきます

新人

- エラーの時は「エラーメッセージ」というものが出ます
- PHPの「それ」は、比較的読みやすいです
  - 「他言語と比較して」程度ですが
- わからないなりに、エラーメッセージを読む癖をつけてみましょう
  - 英語ですが、翻訳ソフトとかありますし、割と単語が限定的なので、なれると「英語が苦手」でもどうにかなります
- よく読むと「あ、この単語、前も見た」「ん? この行でエラー、って言ってる?」などの情報がみえてきます
- あとは質問しましょう(笑

先輩

- はじめの頃は「ここをこう直して」でダイレクトに指定するほうがよいのですが
- なれてきたら、まずは「エラーメッセージを読みながら」説明するとよいです
  - はじめは「どの行でエラーなのか」がどこに書いてあるか
  - 次に「よく見かけるエラーメッセージ」を、読み上げながら
    - Parse error: syntax error とか Warning: Undefined variable とか
- 徐々に「エラーメッセージから、ある程度、間違いが推測できる」ように、丁寧に説明をしてあげるとよいでしょう

独学

- 「エラーメッセージの読み方」自体は、例えば「php よくあるエラー」あたりでググると、解説のブログがいくつか見つかります
- 可能なら「メンターに質問」を、それが無理なら「徐々になれていく」しかないです
  - この辺が「独学」のつらい所ではあります
- IDEのエラーメッセージやAIをうまく活用すると、「完全に一人」よりは少し楽かもしれません

- まずは「最序盤」のポイントを解説していきました
- 少し「プログラミングになれてきた」頃を想定して、お話を次に進めていきましょう

骨格になる文法を学ぶ

- 「何を持って骨格とするのか」ってのも難しくはあるのですが、可能な範囲で可能な限り、文法を理解しておくとよいです
  - 指針の一つに資格試験があります !(ダイレクトマーケティング
- 一方で、文法を「暗記している」必要はあんまりないです
  - 「覚えている」のは便利ですが「忘れた」らググればOKです
- どちらかというと「文法と、それで何ができるかを理解している」ほうが大事です
- そのために、ある程度たくさん書いたら次は「書いた後で、動きを考えてみる」「考えながら書いてみる」で、少しずつ量に「質」を重ねていくとよいでしょう

- 「構造化プログラミング」なんて言葉がありますが
  - 最近あんまり耳に目にしなくなりましたよねぇ
- まず、以下の構文が「だいたいなんとなく書ける」ようになっておくとよいです
  - 変数と代入と四則演算
    - 整数、小数点数、文字列、bool、null
  - 比較演算子とif(else)文とswitch文とmatch式
  - for文とwhile文
  - 配列と連想配列とforeach文
  - コメント
  - 関数(ユーザー定義関数)

先輩

- 「覚えて欲しい文法」は正直たくさんあると思います
- ただ「全部まとめて」教えると、新人さんがパンクします
- 「新しい事は1つになる」ように教えるとよいでしょう
  - この辺は「教える順番の設計」が重要になります
- 例) for文を教えたい
  - for($i = 0; $i < 10; ++$i)
    - 代入式 $i = 0
    - 比較式 $i < 10
    - 加算子 ++$i
    - for文

独学

- まずは書籍なりEラーニングなりを1つ、最後までこなしてみましょう
- その後、できたらあと数冊、数サイト、別筆者の「初心者用教材」で学習をしてみるとよいでしょう
  - 「やったことがある」が多いと思いますが、時々「え? これやってない」があるかと思います
  - 「やったことがある」は重要な構文でもあるので、重複して学んでおいて損はないでしょう

AIを使ってみる

- このあたりはまだ「暗中模索」ではあるかと思うのですが
- 例えば、ChatGPTで以下のプロンプトを渡すと、(難易度はいささかまちまちでしたが)問題を出すことができたので、学習の一助にはなるかもしれません
(カギ括弧の中は目的に併せて入れ替えてください)

  - PHPを学習するために、簡単な試験問題を出してほしいです。"コードを問題に出して出力結果を設問にする"ではなくて"問題は文章だけで、コードを書かせる"ような試験問題をいくつか出してください。
内容は「*関数の書き方について*」がいいです。
難易度は「*専門学校を卒業したて*」くらいでお願いします。

寄り道)プログラム以外も少しやる

- PHPは「Webアプリを作る」のが特に楽な言語です
- ただ「Webアプリ」を作るためには、PHP以外にも以下のあたりは最低限学習をしておくとよいでしょう
  - やるタイミングは「おいおい」でよいかとは思いますが
- RDB
  - 試験的にも必須です。こだわり等がなければ、シェアの高いMySQLをお勧めします
- インフラ
  - 試験的には不要ですが、システム構成としては「必須」なので、多少なり知っておくとメリットがあります
- GitHub
  - よく使われるので、最低限は把握しておきましょう

「なにか」を作ってみる

- そんなに大がかりなものでなくてよいので、なにか1つ2つ(以上)サイトとかシステムとかを作ってみるとよいでしょう
- 目安としては「全体で5画面以内」くらいの、コンパクトなものを作る事をお勧めします
  - 「これが作りたい!」という強いモチベーションがある時は、(先輩なりメンターなりがいれば)作りたいものを「気合いで作る」のも、大変によい勉強になります
- 「なにかを入力して」「計算や判定をして」「出力する」くらいのものがコンパクトで作りやすいです
  - 「BMI計算」「金利計算」「年月日から曜日算出」など

新人

- もし「作ってみたいもの」があるのであれば、迷わずに先輩やメンターに相談してみましょう
  - もしかしたら「結構大変」かもしれませんが、「作ってみたい」のモチベーションはとても大切だと思います
- それ以外であれば、先輩かメンターがいたら、先輩やメンターに相談をしてみましょう

先輩

- 業務にある程度沿ったものが「会社的にはベター」ではあるとは思いますが、難易度の問題もあるので、そこまでこだわらなくてもよいと思います
  - 「社内で使うシステムとか、あるとちょっと便利なツール」あたりは「社内なので割と融通が利く」「率直な意見が返ってくる」ので扱いやすいです
- 「何を学ばせたいのか」をはっきりさせておくと、相手にも伝わりやすいし、学習効果も高いと思います

独学

- もし「作りたいもの」がある場合
  - 難易度の計測ができないので、とりあえず「頑張ってみる」
  - メンターがいるなら速やかに全力で頼る
  - サービスを公開する時はセキュリティに気をつける
    - できるだけ「漏らしちゃいけない情報」はそもそも取得しない
    - クラックされたら「全リセットできるようにする」くらいの準備をする
- もし「作りたいもの」がない場合
  - クラウド系の案件サービスとかを覗き見て「これ、できそうかな?」と思う案件をチョイスして、(受託はせず個人的に)作ってみる
    - 依頼者がいるということは「案件たり得る」内容なので、実践的な内容にはなる

作ったものに仕様変更や追加をする

- 「一回作って作りっぱなし」なシステムはほぼありません
  - お仕事の大半は「既存コードに対する修正や追加」です
- なので「いったん仕上がったコード」に対する修正や追加、という作業は、とても勉強になります
  - 苦労する可能性もいっぱいありますが、それもまた勉強です!!

先輩

- 「社内ツール」を作ってもらってる場合は、せっかくなので「ヒアリング」なども経験してもらいつつ改善点を聞き出して、その中から「比較的簡単そうなもの」を選んであげるとよいでしょう
  - 必要であれば、ヒントやら(設計などの)教育やら、も視野に
- 普通になにか作ってもらっている場合、「比較的ありがちな仕様変更」を想定して、渡してあげるとよいでしょう

独学

- メンターがいたら全力で頼りましょう！
- メンターがいない場合、このあたりの訓練はかなり難しいのですが……
  - 「項目の追加」をしてみる
  - データを一覧する画面があれば
    - sort条件の変更
    - filter条件の変更
    - 表示件数の可変化(1ページ10件と1ページ20件、とか)

- 改修や追加をすると、「比較的楽にいける」こともありますが「結構やっかい」なこともあります
- その辺を楽にする一助が、関数であったり、クラス(後述)であったりします
  - ってのを頭の片隅に入れておくと、関数やクラスの学習がはかどるかと思います

骨格になる文法を学ぶ その２

- 業務においては「クラス」と呼ばれるものを使うことが非常に多いです
  - 個人的には「クラスを使っていない業務コード」は、10年単位で拝見していないです
- 色々な見解があるのですが、以下が、多分一番「身も蓋もない」説明です
  - クラスは「連想配列(構造体)」に「関数」を紐付けたものです
    - 構造体は「keyが固定で決まってる連想配列」くらいに思ってください
- 学習が進むといろいろと突っ込めるようになりますが、それまではこれくらいの解像度でよいと思います

- ただこのクラス、普通に書く以外にも、いろいろあります
  - コンストラクタとデストラクタ
  - 継承
  - インタフェース
  - トレイト
  - static
    - 静的メソッド / 静的プロパティ
    - 遅延静的束縛 (Late Static Bindings)
  - マジックメソッド各種
  - readonly とか プロパティフック とか非対称可視性プロパティ とかレイジーオブジェクト とか 列挙型 とか(PHP8.0ではないので試験的には非対応)

- まぁやりこんでいくときりがないので、いったんは以下くらいをやっておくと(最低限)よいでしょう
  - 普通のクラス(クラス名書いてプロパティ(変数)とメソッド(関数)と定数書いて)
    - アクセス権は把握しておきましょう
  - 継承とインタフェース
  - コンストラクタ

(関数と)クラスは設計が大事!!

- 関数とクラスで本当に大事なのは「どんなふうに切り出すか」です
  - クラス設計、なんて言い方をします
- 先の長いところなので、まずは「言われたとおりに書ける」くらいを目指していくとよいでしょう
- そして「比較的楽にすらすらと書ける」ようになったら、ゆっくりと「設計」に意識を向けていくとよいでしょう
- 設計をする時は、以下を意識しておくとよいでしょう
  - そのクラスの「1インスタンス」は何を表しているのか?
    - これを「短い1文」で説明できるくらいの粒度が、一つの目安です

改めて「質量転換」

- 個人的にはよく「格闘ゲーム」とか「音楽ゲーム」とか「ダンス」とか「舞踊」とかで例えるのですが
  - 「わかってない」と、昇竜拳はそもそも打てない
  - 昇竜拳の打ち方を「理解」してても、戦闘中に入力をミスる
    - 「→で、↓で、↘で、Yボタン」のつもりが「→で、↓で、→で、Yボタン」
  - 体になじませると「無意識」に入力できる
- コードも同じようなところがあります
- なので、まずは「量」書いて、体になじませていきましょう
  - そうしないと、普段「コードを書いている」最中に「とっさに出てこない」技術になってしまいます

「テスト」から「業務」へ

- おそらく、学習のはじめはこんな「設問」でしょう
  - 1から12までの12回のループをfor文で処理するコードを書け
- 業務だと、例えばこんな風な「要求」になります
  - 指定年の月次処理をしてください
- この場合「指定年の月次処理」が「指定年の1月～12月まで」で「指定年で、1から12までのループ」になる、という風に読み解いていく必要があります
- こんな風に「要求の指示」から「具体的な実装の仕方」を推測していく力が求められてきます
  - この辺は「慣れ」と「場数」ですね

高度な書き方

- 学習を初めて少しすると「ある程度なんでも書ける」ようになります……が、そこは「ゴール」ではなく「スタートライン」です
- 同じ動作でも、より見やすかったり簡易だったり安全だったり変更しやすかったりする記法が存在しえます
- また「今」のすべてを把握していたとしても「新しい記法」が追加されることもあります
  - とはいえBcMathの「演算子のオーバロード」はびっくりしました
- 学習は連綿と続くものなので、定期的に再学習をするとよいでしょう
  - PHP マニュアルの「付録」にバージョン毎の差異が載っているので便利です

おまけ)試験対策

- スキルを伸ばす事にもつながるのですが、試験対策についても少し考えてみましょう

書籍を読む

- 1つ有効なのが書籍です
- 主教材である「独習PHP 第4版」や、それ以外でも初心者本を(できれば数冊)読んで、不明瞭なところがないかを確認していきましょう

公式サイトのドキュメントを読む

- 情報としては「最も信頼できる」鉄板の情報です
  - https://www.php.net/manual/ja/
- ただ「初心者に優しく読みやすいか？」と問われると、いささか……
  - とても丁寧に書かれているので「より深く理解する」にはとても適しているのですが
- なので「初心者本を1冊2冊(以上)読んだ後」に読むのをお勧めします

模擬試験や問題集をやってみる

- 模擬試験は、認定スクールであるプライム・ストラテジー株式会社さんが実施されています
  - https://study.prime-strategy.co.jp/study/ph8el1/
- 公式問題集も出ています
  - https://www.amazon.co.jp/dp/B0CVVPTMQL
  - （上級の問題集もでました: ダイレクトマーケティング）
- いずれも「本番の試験と同じフォーマット」なので試験のイメージがつくのと、知識的にも「足りない部分の把握」ができるので、今の自分の実力が判断できます

試験のポイント

- 試験の出題範囲と出題率は公式サイトに載っています
  - https://www.phpexam.jp/summary/novice8
- 中でも、比較的得点配分が高いのが以下です
  - オブジェクト指向構文　17.5%
  - PHPの基本　12.5%
  - 制御構文　　13.8%
  - 演算子　　10.0%
  - リクエスト情報　　10.0%
- これで65%くらい。「基本的な構文」が理解できていればいいところまで狙えます(合格は7割正解)

- 試験問題は実際に「業務で使う範囲」がほとんどなので、「業務で十全にPHPを使っている人」であれば大体、機知の内容です
- なので逆に言うと「試験範囲を十全に把握すると、業務をこなすために必要な足腰がしっかりする」とも言えます

- 初級の後は、是非、上級にも手を出してみてください!
- 上級は「コードを山盛りに読み込む」試験になっています
  - ダイレクトマーケティング

その後は

- 一方では「実際にサイトを作ってみる」のが大切で、よい勉強になります
- しかしもう一方で「常に学び続けていないと」身につかない知識領域もあります
  - ×「必要になってから学習すればいいや」
    - 「必要であることに気づけない」事や知識領域もたくさんあります
    - 例）アルゴリズム、セキュリティ
- マラソンのように「無理なくコンスタントに」学習するのが、スキルアップの秘訣なんじゃないかと思います
  - 「百尺竿頭に一歩を進む」なんて言葉もあります

まとめ

- これから「新しく社員になる」「新しい人を迎える」など、一区切りのある人も多いかと思います
- もちろん「自分のスキルを伸ばしたり」「自分のコードを書いたり」も大切ですが、周囲から教わる、周囲に教えることもまた、大切なことだと思っています

- みなさんの学習が順調に進む事を心より祈念いたします